View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by eScholarship - University of California

UC Davis
IDAV Publications

Title

On Approximating Contours of the Piecewise Trilinear Interpolant Using Triangular Rational-Quadratic Bezier Patches

Permalink

https://escholarship.org/uc/item/7sj7d4n4

Journal

IEEE Transactions on Visualization and Computer Graphics, 3

Authors

Hamann, Bernd
Trotts, Isaac J.
Farin, Gerald

Publication Date

1997

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

On Approximating Contours of the Piecewise Trilinear Interpolant Using Triangular Rational-Quadratic Bézier Patches

Bernd Hamann, *Member, IEEE*, Issac J. Trotts, and Gerald E. Farin

Abstract—Given a three-dimensional (3D) array of function values $F_{i,j,k}$ on a rectilinear grid, the marching cubes (MC) method is the most common technique used for computing a surface triangulation $\mathcal{T}$ approximating a contour (isosurface) $F(x, y, z) = T$. We describe the construction of a C^0-continuous surface consisting of rational-quadratic surface patches interpolating the triangles in $\mathcal{T}$. We determine the Bézier control points of a single rational-quadratic surface patch based on the coordinates of the vertices of the underlying triangle and the gradients and Hessians associated with the vertices.

Index Terms—Approximation, contour, isosurface, marching cubes, rational Bézier curve, rational Bézier surface, triangular patch, triangulation, trilinear interpolation, visualization.

1 Introduction

One of the most commonly used methods to visualize and study a scalar field $F(x, y, z)$ is based on the approximation and rendering of a contour (isosurface) $F = T$. Usually, the function F is evaluated or given on a rectilinear grid, and piecewise trilinear interpolation is used to approximate f inside each cell (cuboid) in the grid. In the following, it is assumed that the function values $F_{i,j,k}$ are given at uniformly spaced points $\mathbf{x}_{i,j,k}$ with integer coordinates, i.e., $\mathbf{x}_{i,j,k} = (i, j, k)^T$. Parametrizing a single cell over the interval [0, 1] in x–, y–, and z–direction and using a local indexing scheme, i.e., denoting the function values at the corners of a particular cell by $f_{0,0,0}, f_{1,0,0}, f_{0,1,0}, \ldots, f_{1,1,1}$, the trilinear interpolant for this cell is given by

$$\begin{aligned} f(x,y,z) = {} & f_{0,0,0}\,(1-x)\,(1-y)\,(1-z) \\ & + f_{1,0,0}\,x\,(1-y)\,(1-z) \\ & + f_{0,1,0}\,(1-x)\,y\,(1-z) \\ & + f_{1,1,0}\,x\,y\,(1-z) \\ & + f_{0,0,1}\,(1-x)\,(1-y)\,z \\ & + f_{1,0,1}\,x\,(1-y)\,z \\ & + f_{0,1,1}\,(1-x)\,(1-y)\,z \\ & + f_{1,1,1}\,x\,y\,z \end{aligned}$$

$$= \sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{1} f_{i,j,k} B_i^1(x) B_j^1(y) B_k^1(z), \qquad (1)$$

where $B_i^1(x)$, $B_j^1(y)$, and $B_k^1(z)$ are Bernstein-Bézier polynomials of degree one, see [11] and [22]. An alternative representation of (1) is given by

$$\begin{aligned} f(x,y,z) &= \sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{1} \Delta^{i,j,k} x^i y^j z^k \\ &= \sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{1} \Delta^{i,j,k} f_{0,0,0} x^i y^j z^k, \end{aligned} \qquad (2)$$

where $\Delta^{i,j,k}$ is the forward difference operator for triple indices, e.g., $\Delta^{0,0,0} f_{0,0,0} = f_{0,0,0}$, $\Delta^{1,0,0} f_{0,0,0} = f_{1,0,0} - f_{0,0,0}$, and $\Delta^{1,1,0} f_{0,0,0} = f_{1,1,0} - f_{1,0,0} - f_{0,1,0} + f_{0,0,0}$.

The MC method computes the intersections of a contour with the edges of a cell based on trilinear interpolation, see [19]. The intersections are then connected to form closed polygons whose edges all lie on certain faces of the cell. The polygons are triangulated, thus defining a piecewise linear approximation to the contour. The original MC method suffers from the fact that the resulting surface triangulations can contain *cracks*, i.e., certain edges in the triangulation might not be shared by exactly two triangles. Various strategies have been proposed to solve this shortcoming, see, e.g., [13] and [21].

The authors are not aware of any research that studies the exact nature of a contour of (1). This paper discusses the construction of an approximation of a contour of (1) using triangular rational-quadratic surface patches. The approximating rational-quadratic surfaces are represented in triangular rational Bézier form, see [9]. In order to model or postprocess an approximation of a contour of a trivariate function, one would prefer a representation in terms of parametric patches. This is the primary motivation for the scheme presented in this paper.

The authors are aware of only one reference dealing with the approximation of a contour by parametric surface

• *B. Hamann and I.J. Trotts are with the Department of Computer Science, University of California, Davis, CA 95616-8562. E-mail: {hamann, trotts}@cs.ucdavis.edu.*
• *G. Farin is with the Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287-5406. E-mail: farin@asu.edu.*

For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number 105373.

1077-2626/97/$10.00 © 1997 IEEE

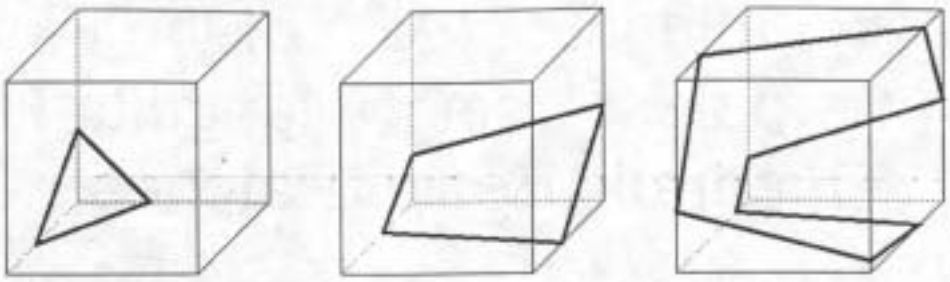

Fig. 1. Polygons obtained by MC method using *asymptotic decider* described in [21].

patches. This reference is [12]. The construction described in [12] yields patches with three, four, five, and six boundary curves. The topology at vertices is characterized by the fact that exactly four patch boundary curves meet at a shared patch corner vertex, i.e., each patch corner vertex has valence four. Furthermore, not all vertices produced by an MC algorithm are interpolated by their method. Our method does not require certain valences for patch corner vertices, and it interpolates all the points generated by an MC algorithm. We plan to develop an algorithm that can be used to reduce the number of triangular patches, based on the idea of identifying patches in (nearly) planar regions and eliminating those patches.

2 Outline of an MC Method Based on Rational-Quadratic Surfaces

The fact that the contour of a bilinear interpolant is a hyperbola is used for the construction of the edges of the closed polygons whose vertices lie on the edges of a grid cell and on a particular contour. In general, one can obtain closed polygons with three, four, five, six, seven, eight, nine, or 12 vertices, see [13]. Examples are shown in Fig. 1.

These closed polygons are then triangulated defining a coarse, piecewise, linear approximation of the contour inside the grid cell. The triangulation process itself must consider various special cases, including certain cases where additional points on the contour in the interior of a cell must be added to generate topologically valid triangulations. Again, this is discussed in [21]. In this paper, we propose an alternative approach for the representation of a contour approximation inside a grid cell. This alternative approach uses rational-quadratic surface patches.

Triangular quadric surface patches in Bézier representation are discussed in [2], [3], [16], [17], and [20]. When approximating a contour of the trilinear interpolant by a rational-quadratic surface, the construction of the various triangular rational Bézier patches is rather straightforward: The closed polygons defined by points on the edges of a grid cell are triangulated, and the edges in the resulting triangulations are used in the construction of patch boundary curves, which completely define a set of triangular Bézier patches in the interior of the grid cell. These patches approximate the contour locally. Overall, the patches associated with a particular cell are constructed by performing these steps:

Step 1—*Triangulation.* Given the set of all closed polygons with edges on the faces of a grid cell, these polygons are triangulated.

Step 2—*Boundary curves on faces.* The edges in the triangulations which lie on the faces of a grid cell can be viewed as linear approximations of hyperbolic arcs; the exact rational Bézier representation of all these arcs on the grid cell's faces are determined.

Step 3—*Boundary curves in interior.* The edges of the triangulations which lie in the interior of a grid cell are used for the construction of patch boundary curves (conics): First, planes are constructed that will contain the patch boundary curves; second, conic approximations in rational Bézier representation of all patch boundary curves lying in the interior of the grid cell are determined.

Step 4—*Triangular Bézier patches.* The conics resulting from Steps 2 and 3 completely define the analytical approximation of a contour of the trilinear interpolant associated with a grid cell; they are used for the definition of the control nets of triangular rational-quadratic Bézier patches.

Fig. 2 illustrates these four steps, which are discussed in detail in Sections 4 and 5. In Section 3, we address issues related to the general nature of a contour of the trilinear interpolant.

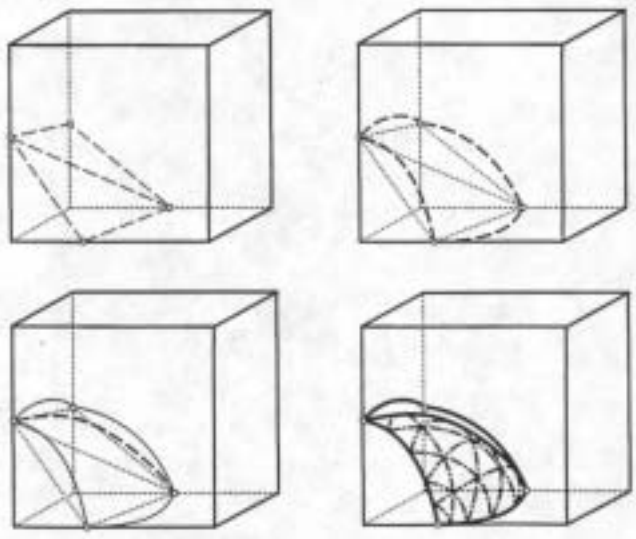

Fig. 2. Triangulation and boundary curve and patch construction.

3 The Rational Bilinear Patch and Contours of the Trilinear Interpolant

In [10], a parametric rational bilinear patch $\mathbf{x}(u, v)$ defined over $[0, 1] \times [0, 1]$ is represented as

$$\mathbf{x}(u,v) = \frac{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}\mathbf{b}_{i,j}B_i^1(u)B_j^1(v)}{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}B_i^1(u)B_j^1(v)}. \quad (3)$$

Here, $\mathbf{b}_{i,j}$ denotes a Bézier control point, and $\omega_{i,j}$ denotes its associated weight. The rational bilinear patch is characterized by these facts:

1) Any planar section of the rational bilinear patch is a conic.
2) The rational bilinear patch is a quadric.
3) The rational bilinear patch is either a hyperbolic paraboloid or a hyperboloid of one sheet.

Using the alternative representation (2) of the trilinear interpolant (1) and solving for z, the contour $f(x, y, z) = T$ can be written as

$$z(x,y) = \frac{T - \Delta^{0,0,0} - \Delta^{1,0,0}x - \Delta^{0,1,0}y - \Delta^{1,1,0}xy}{\Delta^{0,0,1} - \Delta^{1,0,1}x - \Delta^{0,1,1}y - \Delta^{1,1,1}xy}. \quad (4)$$

Converting the numerator and denominator in (4) to Bernstein-Bézier form and using the rules for dividing a Bernstein-Bézier surface by a (scalar-valued) Bernstein-Bézier polynomial of the same degree, see [23], (4) can be rewritten in (nonparametric) Bernstein-Bézier form as

$$z(x,y) = \frac{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}\mathbf{b}_{i,j}B_i^1(x)B_j^1(y)}{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}B_i^1(x)B_j^1(y)}. \quad (5)$$

The necessary substitutions are

$$\begin{aligned}
b_{0,0} &= \frac{T - \Delta^{0,0,0}}{\Delta^{0,0,1}},\\
b_{1,0} &= \frac{T - \Delta^{0,0,0} - \Delta^{1,0,0}}{\Delta^{0,0,1} - \Delta^{1,0,1}},\\
b_{0,1} &= \frac{T - \Delta^{0,0,0} - \Delta^{0,1,0}}{\Delta^{0,0,1} - \Delta^{0,1,1}},\\
b_{1,1} &= \frac{T - \Delta^{0,0,0} - \Delta^{1,0,0} - \Delta^{0,1,0} - \Delta^{1,1,0}}{\Delta^{0,0,1} - \Delta^{1,0,1} - \Delta^{0,1,1} - \Delta^{1,1,1}},\\
\omega_{0,0} &= \Delta^{0,0,1},\\
\omega_{1,0} &= \Delta^{0,0,1} + \Delta^{1,0,1},\\
\omega_{0,1} &= \Delta^{0,0,1} + \Delta^{0,1,1}, \text{and}\\
\omega_{1,1} &= \Delta^{0,0,1} + \Delta^{1,0,1} + \Delta^{0,1,1} + \Delta^{1,1,1}.
\end{aligned}$$

Formally, (5) is a special, nonparametric case of (3). The weights in this representation can be negative. This causes problems for the evaluation (poles), e.g., the Bézier ordinates and weights implied by the eight corner data $f_{0,0,0} = f_{1,1,0} = f_{1,0,1} = f_{0,1,1} = 0$, $f_{1,0,0} = f_{0,1,0} = f_{0,0,1} = f_{1,1,1} = 1$, and $T = 0.55$ are $b_{0,0} = b_{1,1} = 0.55$, $b_{1,0} = b_{0,1} = 0.45$, $\omega_{0,0} = \omega_{1,1} = 1$, and $\omega_{1,0} = \omega_{0,1} = -1$. This particular example is rendered in Fig. 3 Rewriting (5) in parametric form yields

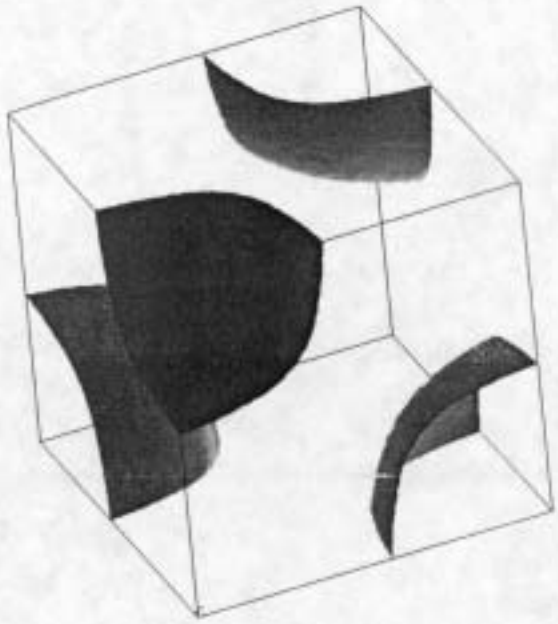

Fig. 3. The four components of the cubic surface defined by contour $f = 0.55$; $f_{0,0,0} = f_{1,1,0} = f_{1,0,1} = f_{0,1,1} = 0$ and $f_{1,0,0} = f_{0,1,0} = f_{0,0,1} = f_{1,1,1} = 1$.

$$\mathbf{x}(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} = \begin{pmatrix} u \\ v \\ \dfrac{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}b_{i,j}B_i^1(u)B_j(v)}{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}B_i^1(u)B_j^1(v)} \end{pmatrix}$$

$$= \frac{\sum_{i=0}^{1}\sum_{j=0}^{1} i\, B_i^1(u)B_j^1(v)}{\sum_{i=0}^{1}\sum_{j=0}^{1} j\, B_i^1(u)B_j^1(v)} \quad (6)$$

$$= \frac{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}b_{i,j}B_i^1(u)B_j(v)}{\sum_{i=0}^{1}\sum_{j=0}^{1} \omega_{i,j}B_i^1(u)B_j^1(v)},$$

which *is not* a parametric rational bilinear patch of the form defined by (3). Due to the fact that (1) contains the cubic term $f_{1,1,1}\, x\, y\, z$, the contour $f = T$ is a *cubic* surface and, using rational-quadratic surface patches, we can only *approximate* the contour.

In the following, we describe the construction of a contour approximation inside a grid cell by means of triangular rational-quadratic Bézier patches. An in-depth discussion of conic sections, quadric surfaces, their relation to projective geometry, and a bibliography regarding these topics can be found in [4].

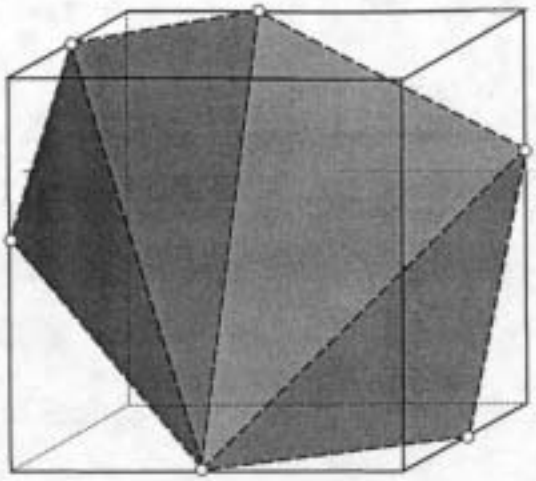

Fig. 4. Triangulation of six-sided polygon.

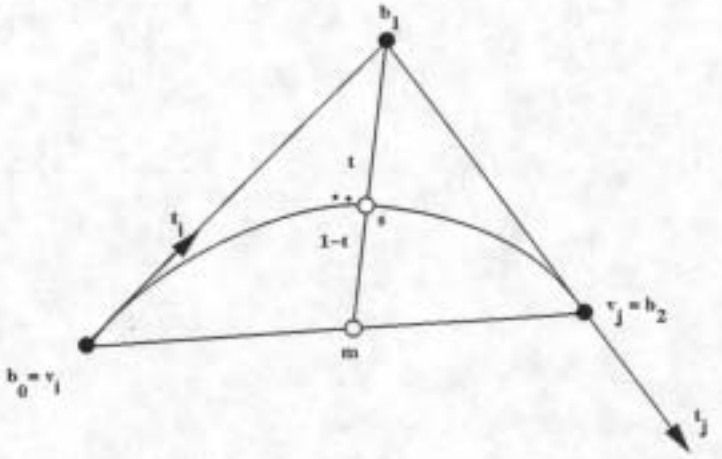

Fig. 5. Construction of Bézier point $\mathbf{b}_1$ and its weight ω_1 for hyperbolic arc on cell's face.

4 Construction of the Triangular Patches Approximating the Contour

According to the sequence of Steps 1-4 described in Section 2, all closed polygons with edges on the faces of a grid cell must be triangulated. This paper is not concerned with the construction of "good triangulations" for such polygons. For the purposes of this paper, it is sufficient to use a very simple triangulation strategy: Denoting the (ordered) vertices of a closed polygon by $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots, \mathbf{v}_N, N \in \{3,4,5,6,7,8,9,12\}$, one possible triangulation is defined by the vertex triples $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}, \{\mathbf{v}_1, \mathbf{v}_3, \mathbf{v}_4\}, \{\mathbf{v}_1, \mathbf{v}_4, \mathbf{v}_5\}, \ldots$, and $\{\mathbf{v}_1, \mathbf{v}_{N-1}, \mathbf{v}_N\}$. Using this triangulation strategy, all triangles—associated with this particular polygon—have the same orientation. An example is shown in Fig. 4.

The triangulation strategy has an influence on the final contour approximation. We have not yet investigated which triangulation strategy produces "best" final approximations. Various criteria regarding the quality of triangulations in the plane are discussed in [24]. Most of these criteria can easily be generalized for surface triangulations in three dimensions. A quality criterion that seems appropriate for our application could relate triangulation quality to the variation of the normals of all pairs of triangles sharing a common edge, see [5]. Several methods are known to improve triangulations—by performing local optimization procedures. A commonly used method is described in [18]; the method is based on swapping diagonals of quadrilaterals (defined by pairs of neighboring triangles sharing a common edge) until one can no longer improve a certain triangulation quality measure.

4.1 Patch Boundary Curve Construction on a Face of a Grid Cell

Only the edges $\overline{\mathbf{v}_1\mathbf{v}_2}$, $\overline{\mathbf{v}_2\mathbf{v}_3}$, $\overline{\mathbf{v}_3\mathbf{v}_4}$, ..., and $\overline{\mathbf{v}_N\mathbf{v}_1}$, lie on the faces of the grid cell. All the other edges lie in the interior of the cell. For each edge $\overline{\mathbf{v}_i\mathbf{v}_j}$ lying on a cell face, we determine the rational Bézier representation of the hyperbolic arc passing through $\mathbf{v}_i$ and $\mathbf{v}_j$ and lying in the plane defined by the cell's face, see [21]. In the following, we represent a conic section $\mathbf{c}(t)$, $t \in [0, 1]$, in standard form, i.e.,

$$\mathbf{c}(t) = \frac{\mathbf{b}_0 B_0^2(t) + \omega_1 \mathbf{b}_1 B_1^2(t) + \mathbf{b}_2 B_2^2(t)}{B_0^2(t) + \omega_1 B_1^2(t) + B_2^2(t)}, \tag{7}$$

see [10], [11].

The Bézier points and their associated weights are computed in a straightforward manner. First, we define $\mathbf{b}_0 = \mathbf{v}_i, \mathbf{b}_2 = \mathbf{v}_j$, and $\omega_0 = \omega_2 = 1$. Second, we compute the middle Bézier point $\mathbf{b}_1$ as the intersection point of the lines passing through $\mathbf{v}_i$ and $\mathbf{v}_j$ whose directions are defined by the (normalized) tangent vectors $\mathbf{t}_i$ (at $\mathbf{v}_i$) and $\mathbf{t}_j$ (at $\mathbf{v}_j$). The vectors $\mathbf{t}_i$ and $\mathbf{t}_j$, on the other hand, are defined by the normals of the hyperbola at $\mathbf{v}_i$ and $\mathbf{v}_j$. The normals are given by the gradient of the bilinear interpolant implied by the function values at the corners of the face. Denoting the mid point of the edge $\overline{\mathbf{v}_i\mathbf{v}_j}$ by $\mathbf{m}$, the line $\overline{\mathbf{b}_1\mathbf{m}}$ intersects the hyperbolic arc in the *shoulder point* $\mathbf{s} = (1 - \gamma)\,\mathbf{b}_1 + \gamma\mathbf{m}$, $0 \le \gamma \le 1$. Thus, the weight for $\mathbf{b}_1$ is given by the ratio

$$\omega_1 = \frac{1-\gamma}{\gamma}, \tag{8}$$

see [10], [11] for a derivation of (8). The shoulder point $\mathbf{s}$ is computed by inserting the parametric line equation $(1 - \gamma)\mathbf{b}_1 + \gamma\mathbf{m}$ into the implicit representation of the hyperbola on the particular face. Special cases that can arise are discussed in Section 5 of this paper. This construction assumes that the hyperbolic arc is parametrized over the interval [0, 1]. Fig. 5 illustrates the construction.

Remark 4.1. The notation $\overline{\mathbf{v}_i\mathbf{v}_j}$ implies that this line segment has a direction: It starts at $\mathbf{v}_i$ and ends at $\mathbf{v}_j$. The angles between $\mathbf{t}_i$ and $\mathbf{v}_j - \mathbf{v}_i$ and between $\mathbf{t}_j$ and $\mathbf{v}_j - \mathbf{v}_i$ must not exceed 90 degrees. If these conditions are violated, the sign(s) of the tangent vector(s) is (are) inverted. Possible directions for tangent vectors are shown in Fig. 6

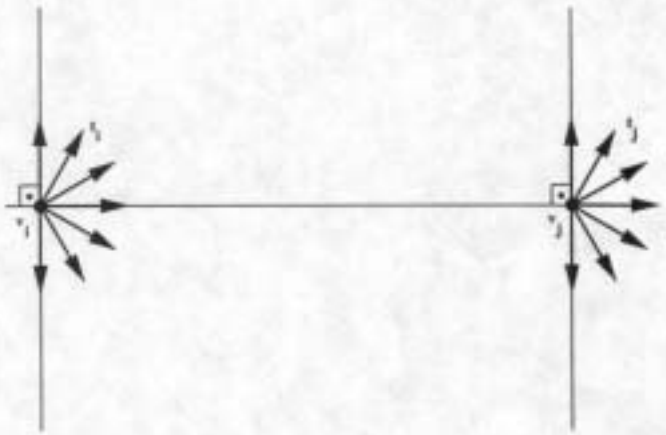

Fig. 6. Possible tangent vectors at end points of directed line segment $\overline{\mathbf{v}_i\mathbf{v}_j}$.

4.2 Patch Boundary Curve Construction in the Interior of a Grid Cell

A more complicated construction is necessary for the computation of the middle Bézier points of the conic sections associated with edges in the triangulation that lie in the interior of a grid cell. The construction described in the following utilizes principles that are similar to ideas discussed in [15], [16], and [17]. Each edge in a cell's interior is shared by exactly two triangles. Denoting the unit (outward) normal vectors of the two triangles sharing an interior edge $\overline{\mathbf{v}_i\mathbf{v}_j}$ by $\mathbf{n}_1$ and $\mathbf{n}_2$, their average, $\frac{1}{2}(\mathbf{n}_1 + \mathbf{n}_2)$, is used to define a plane P that contains $\mathbf{v}_i$ and $\mathbf{v}_j$ and has a normal that is perpendicular to $\frac{1}{2}(\mathbf{n}_1 + \mathbf{n}_2)$. Fig. 7 illustrates the construction of the plane P and the middle Bézier point $\mathbf{b}_1$ used to define an approximating conic section associated with an edge in the cell's interior.

Once the plane P is determined, the principle used to compute the middle Bézier point $\mathbf{b}_1$ of the conic section lying in this plane—approximating the contour of the trilinear interpolant in this plane—differs from the principle used to define the hyperbolic arcs on the cell's faces. The location of the middle Bézier point is still determined by the intersection of two tangents passing through $\mathbf{v}_i$ and $\mathbf{v}_j$, but the computation of the two tangent vectors and the definition of the weight of the middle Bézier point are different.

The two tangent vectors are computed as follows: Considering the fact that the gradient of the trilinear interpolant is perpendicular to the contour, we compute the intersections between P and the two tangent planes (defined by the gradient) of the contour passing through $\mathbf{v}_i$ and $\mathbf{v}_j$; the intersections are two lines $\mathbf{l}_i$ and $\mathbf{l}_j$ which are used to define the tangent vectors (lying in P) at $\mathbf{v}_i$ and $\mathbf{v}_j$. We can now compute the location of the middle Bézier point by intersecting $\mathbf{l}_i$ and $\mathbf{l}_j$ in P.

Due to the fact that, in general, the plane P is not parallel to any of the cell's faces, one can only determine an approximating boundary curve in P. The plane P defines a planar section of the contour. Considering this planar section only, we choose the middle weight according to the following principle: Denoting the exact curvatures at $\mathbf{v}_i$ and $\mathbf{v}_j$ by κ_i and κ_j, the weight for the middle Bézier point is

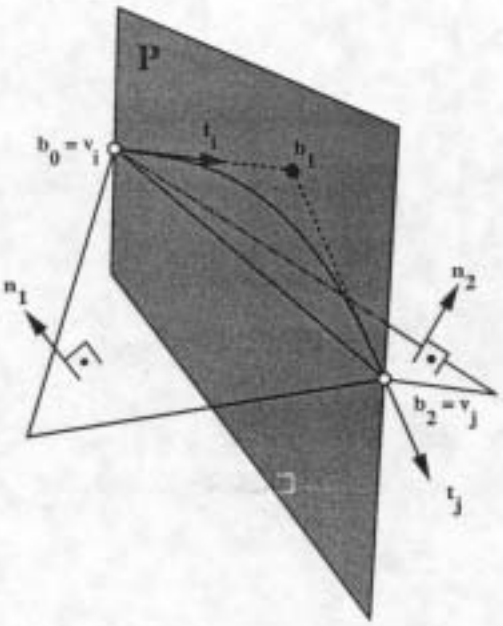

Fig. 7. Construction of plane P and Bézier point $\mathbf{b}_1$ for conic section in interior of grid cell.

chosen such that the approximating conic's curvatures at the end points $\mathbf{v}_i$ and $\mathbf{v}_j$ closely approximate the exact ones. The absolute curvatures of the conic section at its two end points, corresponding to the parameter values $t = 0$ and $t = 1$, are given by

$$\kappa(0) = \frac{A}{\omega_1^2\, l_{0,1}^3} \text{ and } \kappa(1) = \frac{A}{\omega_1^2\, l_{1,2}^3} \tag{9}$$

where A is the area of the triangle with vertices $\mathbf{b}_0$, $\mathbf{b}_1$, and $\mathbf{b}_2$, ω_1 is the weight of $\mathbf{b}_1$, and $l_{i,j}$ is the Euclidean distance between $\mathbf{b}_i$ and $\mathbf{b}_j$. See [10] for a derivation of (9). Solving the two equations in (9) for ω_1, one obtains the two expressions

$$|\omega_1| = \sqrt{\frac{A}{|\kappa(0)|\, l_{0,1}^3}} \text{ and } |\omega_1| = \sqrt{\frac{A}{|\kappa(1)|\, l_{1,2}^3}}. \tag{10}$$

The exact curvatures at the end points $\mathbf{v}_i$ and $\mathbf{v}_j$, as implied by the contour $f = T$ restricted to the plane P, are denoted by κ_i and κ_j, respectively. Their computation is described in the Appendix. The signs of κ_i and κ_j can, in principle, be positive or negative. In the following, we view the conic section to be constructed as a convex curve with *positive* curvatures at its end points. Therefore, we assume that the curve implied by the end points $\mathbf{v}_i$ and $\mathbf{v}_j$ and the associated tangent vectors $\mathbf{t}_i$ and $\mathbf{t}_j$ is convex, too. If this is not the case, we have to use the subdivision scheme for nonconvex data that is discussed in Section 5. Using a *least squares approximation* approach, we define the weight ω_1 of $\mathbf{b}_1$ as

$$\omega_1 = \frac{1}{2}\left(\sqrt{\frac{A}{|\kappa_i|\, l_{0,1}^3}} + \sqrt{\frac{A}{|\kappa_j|\, l_{1,2}^3}}\right). \tag{11}$$

It must be made clear again that the conic section associated with an interior edge is only an approximation of the intersection of the contour with the plane containing the

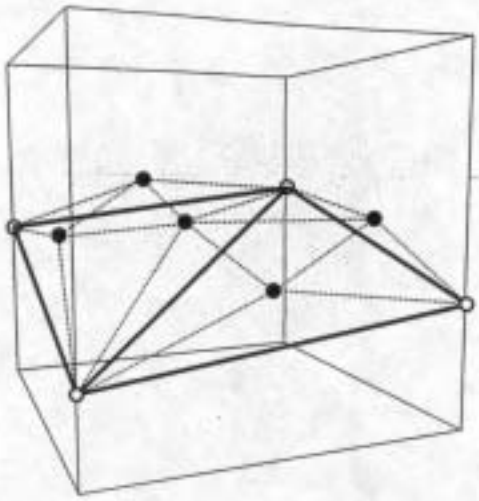

Fig. 8. Triangular Bézier control nets associated with four-sided polygon.

conic. If one wants an exact representation of this boundary curve, one must use higher-degree rational curves. Therefore, one also has to use higher-degree triangular rational patches. Our scheme will lead to tangent plane discontinuities along shared patch boundary curves that lie on cell faces, even if one uses higher-degree rational curves in the interior of a particular cell. Therefore, we have chosen to use rational-quadratic patches everywhere and not to use higher-degree patches. We want to investigate this issue further in the future.

At this point, we have computed the rational Bézier representation of all conic sections associated with the edges in the triangulation of a closed polygon. These conic sections completely determine the triangular rational Bézier patches whose union approximates that part of a contour of the trilinear interpolant that is clipped against the faces of a particular grid cell. An example of the resulting patch control nets is shown in Fig. 8 for a four-sided polygon.

5 Special Cases

There are several special, degenerate cases that must be considered. They are discussed in this section. Cases 1-6 address problems related to the actual degree of the trilinear interpolant.

Case 1—*Constant function on face.* The four function values at the corners of a cell's face are equal. If the values equal T, the cell face is represented by two planar triangular rational-quadratic Bézier patches.

Case 2—*Constant function.* All eight function values at the corners of a cell are equal. If the values equal T, the six faces of the cell are represented by 12 planar triangular rational-quadratic Bézier patches.

Case 3—*Linear function on face.* The four function values at the corners of a cell's face imply a function that varies linearly on the face. If the contour $f = T$ intersects the face in two different points along two of its four edges, the resulting line segment is represented as a conic.

Case 4—*Linear function.* The eight function values at the cell corners imply a function that is linear in x, y, and z. If the contour $f = T$ of this linear function intersects edges of the grid cell, the resulting closed (planar) polygon is triangulated, and each triangle is represented as a (planar) triangular rational-quadratic Bézier patch.

Case 5—*Contour coincides with asymptotes on face.* The asymptotes of a contour of the bilinear interpolant $f(x, y) = \sum_{i=0}^{1}\sum_{j=0}^{1}\Delta^{i,j}x^i y^j$ are the two (orthogonal) lines

$$x_{asy} = -\frac{\Delta^{0,1}}{\Delta^{1,1}} \text{ and } y_{asy} = -\frac{\Delta^{1,0}}{\Delta^{1,1}}, \tag{12}$$

$\Delta^{1,1} \neq 0$, see [21]. On a face of a grid cell the hyperbolic arcs degenerate to these lines whenever the contour value T equals the value of the bilinear interpolant on these two lines (the bilinear interpolant is constant on this pair of orthogonal lines). In this case, *two* line segments are used as boundary curves for the triangular rational-quadratic Bézier patches to be constructed. The two line segments are defined by the lines $x = x_{asy}$ and $y = y_{asy}$.

Case 6—*Contour coincides with asymptotes.* The asymptotes of a contour of the trilinear interpolant (2) are the three (mutually orthogonal) planes

$$x_{asy} = -\frac{\Delta^{0,1,1}}{\Delta^{1,1,1}},\ y_{asy} = -\frac{\Delta^{1,0,1}}{\Delta^{1,1,1}}, \text{ and } z_{asy} = -\frac{\Delta^{1,1,0}}{\Delta^{1,1,1}}, \tag{13}$$

$\Delta^{1,1,1} \neq 0$. A contour degenerates to these planes when the value of T equals the value of the trilinear interpolant in these planes. In general, the trilinear interpolant varies linearly in these three planes, which can be verified by inserting x_{asy}, y_{asy}, and z_{asy} into (2). If a contour degenerates to these planes, we use planar triangular rational-quadratic Bézier patches as representation.

Cases 7-9 are related to the boundary curve construction described in Section 4. They deal with non-convex and a cure for "overshooting" boundary curves, i.e., boundary curves that deviate significantly from a straight line.

Case 7—*Nonconvex boundary curve.* An interior edge $\overline{\mathbf{v}_i\mathbf{v}_j}$ in the triangulation of a closed polygon subdivides the plane P containing the boundary conic into two half spaces: One half space lies to the left of the directed line defined by $\overline{\mathbf{v}_i\mathbf{v}_j}$, and the other half space lies to the right of this line. A nonconvex boundary curve is implied by the data when the two tangent vectors $\mathbf{t}_i$ and $\mathbf{t}_j$ point into the same half space. In this case, the boundary curve will be represented by two conic sections.

First, we reflect the tangent vectors $\mathbf{t}_i$ and $\mathbf{t}_j$ with respect to the axis passing through $\mathbf{v}_i$ and $\mathbf{v}_j$, denoting the reflected vectors by $\mathbf{t}_i^*$ and $\mathbf{t}_j^*$. Next, we compute the intersection points $\mathbf{i}_1$ between the lines $\mathbf{v}_i + s\mathbf{t}_i$ and $\mathbf{v}_j + t\mathbf{t}_j^*$ and $\mathbf{i}_2$ between the lines $\mathbf{v}_i + u\mathbf{t}_i^*$ and $\mathbf{v}_j + v\mathbf{t}_j$. We define the middle Bézier points of the two conic sections as $\mathbf{i}_1$ and $\mathbf{i}_2$, respectively, and use the average of the two intersection points as break point of the

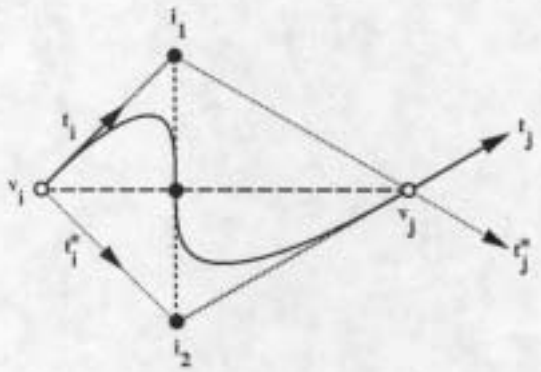

Fig. 9. Construction of nonconvex boundary curve.

two conic sections. This construction also works for parallel end tangent vectors, i.e., $\mathbf{t}_i = \pm\, \mathbf{t}_j$, as long as they are not both perpendicular or parallel to $(\mathbf{v}_j - \mathbf{v}_i)$. Other solutions to this problem are described in [15] and [16]. The construction is illustrated in Fig. 9.

The weights of the two middle Bézier points of the two conic sections are determined as follows: Knowing the exact curvatures κ_i at $\mathbf{v}_i$ and κ_j at $\mathbf{v}_j$ (see Section 4), one computes the middle weight of the first conic (passing through $\mathbf{v}_i$) by requiring that it must interpolate κ_i at $\mathbf{v}_i$ and the middle weight of the second conic (passing through $\mathbf{v}_j$) by requiring that it must interpolate κ_j at $\mathbf{v}_j$. Since the break point of the two sections is, in general, not a point on the contour, we do not care about the implied curvature at the break point of the two sections.

Whenever a nonconvex boundary curve is split in this fashion, one must also split the triangular patch into multiple triangular patches. One must consider three cases:

1) One boundary curve is nonconvex and is split into two conic sections;
2) Two boundary curves are nonconvex and are each split into two conic sections; and
3) All three boundary curves are nonconvex and are each split into two conic sections.

We will consider only Case 1) in more detail. The other two cases require similar treatments. Assuming that the only nonconvex boundary curve is associated with the edge $\overline{\mathbf{v}_i\mathbf{v}_j}$, the original triangle with vertices $\mathbf{v}_i$, $\mathbf{v}_j$, and $\mathbf{v}_k$ is split into two subtriangles. Denoting the break point of the one boundary curve consisting of two conic sections by $\mathbf{m}$, an interior patch boundary curve (also a conic section) is constructed in the plane P that contains the two points $\mathbf{v}_k$ and $\mathbf{m}$ and is perpendicular to the triangle passing through $\mathbf{v}_i$, $\mathbf{v}_j$ and $\mathbf{v}_k$. The normal of the interior patch boundary curve at $\mathbf{v}_k$ is defined by projecting the original normal $\mathbf{n}_k$ into the plane P. We define the first and last Bézier points of this curve as $\mathbf{v}_k$ and $\mathbf{m}$ and the middle Bézier point as

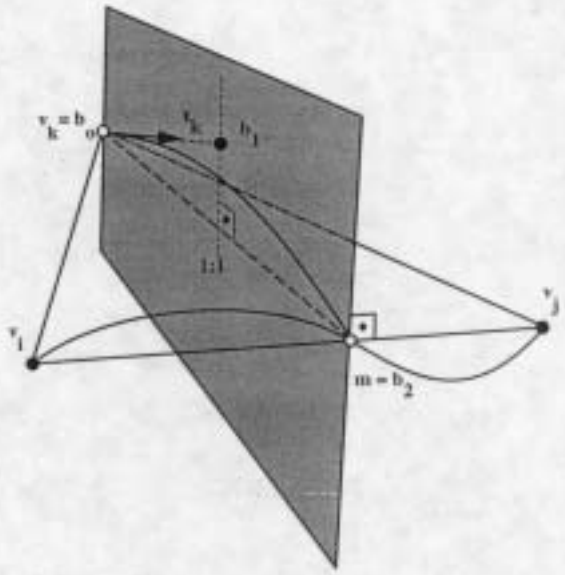

Fig. 10. Construction of interior patch boundary curve.

$$\mathbf{v}_k + \frac{1}{2}\frac{\|\mathbf{m}-\mathbf{v}_k\|^2}{\mathbf{t}_k\cdot(\mathbf{m}-\mathbf{v}_k)}\mathbf{t}_k, \tag{14}$$

where $\mathbf{t}_k$ is the tangent vector of the curve in P at $\mathbf{v}_k$. This means that the perpendicular projection of the middle Bézier point onto the line $\overline{\mathbf{v}_k\mathbf{m}}$ is the midpoint of it. "Overshooting" can be controlled by using the scheme described under Case 9 (see below). If $\mathbf{t}_k$ forms a right angle with $(\mathbf{m}-\mathbf{v}_k)$, we define the middle Bézier point as $\mathbf{v}_k + \frac{1}{2}\|\mathbf{m}-\mathbf{v}_k\|\mathbf{t}_k$. The given (precomputed) curvature value at $\mathbf{v}_k$ defines the weight for the middle Bézier point. Fig. 10 illustrates this construction.

Cases 2) and 3) require the generation of three and four triangular patches. Assuming that the two triangle edges $\overline{\mathbf{v}_i\mathbf{v}_j}$ and $\overline{\mathbf{v}_j\mathbf{v}_k}$ both imply nonconvex boundary curves, we construct a first interior patch boundary curve for the edge $\overline{\mathbf{v}_i\mathbf{m}_{j,k}}$ (see previous paragraph) and a second one for the edge $\overline{\mathbf{m}_{i,j}\mathbf{m}_{j,k}}$. Here, $\mathbf{m}_{i,j}$ denotes the break point of a curve consisting of two conic sections passing through $\mathbf{v}_i$ and $\mathbf{v}_j$, respectively. We choose the line segment $\overline{\mathbf{m}_{i,j}\mathbf{m}_{j,k}}$ as the second interior patch boundary curve. Regarding Case 3), we choose the three interior patch boundary curves to be the line segments $\overline{\mathbf{m}_{i,j}\mathbf{m}_{j,k}}$, $\overline{\mathbf{m}_{i,j}\mathbf{m}_{i,k}}$, and $\overline{\mathbf{m}_{i,k}\mathbf{m}_{j,k}}$, which leads to a planar triangular patch "in the center."

REMARK 5.1. A boundary curve degenerates to a straight line whenever $\mathbf{t}_i$ or $\mathbf{t}_j$ has the same direction as $(\mathbf{v}_j - \mathbf{v}_i)$.

REMARK 5.2. It is also possible to use *degree-elevated* conics, i.e., rational cubic curves, as boundary curves of the triangular patches. The concept allows to handle nonconvex boundary curves more easily, see [15]. We do not use this concept here since

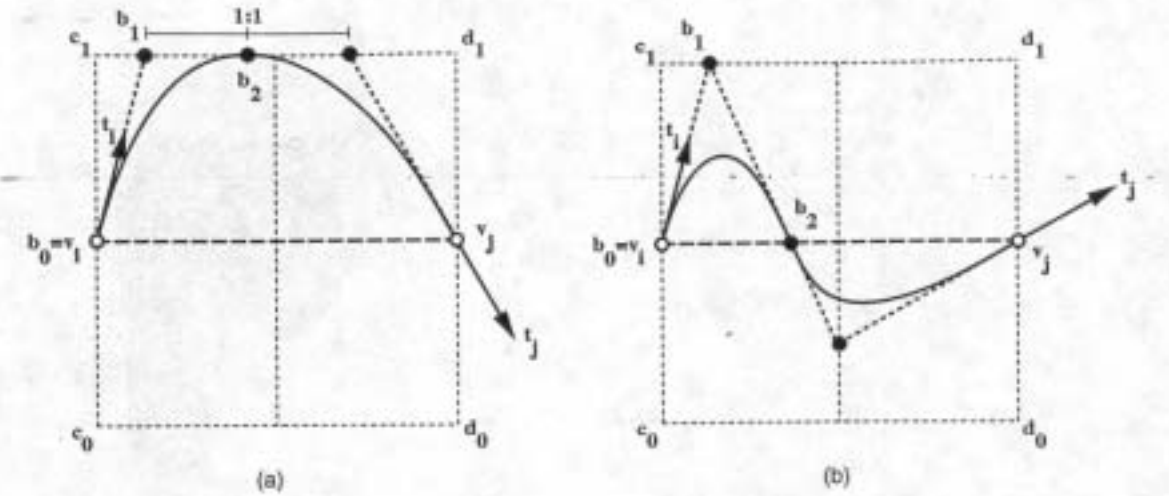

Fig. 11. Avoiding undesired "overshooting" of (a) convex curve and (b) nonconvex curve, using splitting.

1) the number of control points per patch would increase from six to 10 and
2) the computation of the center Bézier point $\mathbf{b}_{1,1,1}$ of each triangular patch would cause additional problems.

REMARK 5.3. The construction of the interior patch curves resulting from subdivision and emanating from (or ending at) one of the break points $\mathbf{m}_{i,j}$, $\mathbf{m}_{j,k}$, or $\mathbf{m}_{i,k}$ ensures that patch subdivision terminates. Interior patch curves are are either convex curves or line segments that do not require further splitting.

REMARK 5.4. When applying the patch subdivision process, one obtains convex boundary curves (including line segments) only, being the shared boundary curves of the resulting subpatches. Thus, the process terminates after one level of subdivision.

Case 8—*Horizontal tangents.* If both end tangent vectors $\mathbf{t}_i$ and $\mathbf{t}_j$ are (positive) multiples of $(\mathbf{v}_j - \mathbf{v}_i)$, the boundary curve becomes a straight line segment with control points $\mathbf{v}_i$, $\frac{1}{2}(\mathbf{v}_i + \mathbf{v}_j)$, and $\mathbf{v}_j$.

Case 9—*Avoiding "overshooting" boundary curves.* Following the boundary curve construction as described above can lead to curves that highly "overshoot" for certain tangent conditions at the end points, for convex and non-convex boundary curves. For example, whenever the two vectors $\mathbf{t}_i$ and $\mathbf{t}_j$ are nearly perpendicular to $(\mathbf{v}_j - \mathbf{v}_i)$—but not parallel to each other—the resulting conic section(s) will deviate very much from the line segment $\overline{\mathbf{v}_i\mathbf{v}_j}$.

The degree of "overshooting" can be controlled by restricting the locus of a boundary curve by forcing it to lie inside the square region R with the four corner points $\mathbf{c}_0 = \mathbf{v}_i - \frac{1}{2}l\mathbf{v}^{\perp}_{i,j}$, $\mathbf{c}_1 = \mathbf{v}_i - \frac{1}{2}l\mathbf{v}^{\perp}_{i,j}$, $\mathbf{d}_0 = \mathbf{v}_j - \frac{1}{2}l\mathbf{v}^{\perp}_{i,j}$, and $\mathbf{d}_1 = \mathbf{v}_j - \frac{1}{2}l\mathbf{v}^{\perp}_{i,j}$, where $l = \|\mathbf{v}_j - \mathbf{v}_i\|$ and $\mathbf{v}^{\perp}_{i,j}$ is a unit vector that is normal to $(\mathbf{v}_j - \mathbf{v}_i)$. Usually, one would like to avoid

1) overshooting convex patch boundary curves;
2) overshooting nonconvex patch boundary curves; and
3) overshooting interior patch curves resulting from patch subdivision.

9A Avoiding Overshooting Convex Patch Boundary Curves

Whenever the default construction of a convex boundary curve leads to a conic section partially lying outside the region R, the curve will be represented by two curve segments that both lie inside R. To test whether a conic section lies outside R we simply test whether its shoulder point lies outside, see [10].

Denoting the control points of a conic section that partially lies outside R by $\mathbf{b}_0$, $\mathbf{b}_1$, and $\mathbf{b}_2$, we compute the intersection points $\mathbf{i}_1$ between the line segment $\overline{\mathbf{b}_0\mathbf{b}_1}$ and the boundary of R and $\mathbf{i}_2$ between the line segment $\overline{\mathbf{b}_1\mathbf{b}_2}$ and the boundary of R. We define the Bézier control points of the first segment as $\mathbf{b}_0$, $\mathbf{i}_1$, and $\frac{1}{2}(\mathbf{i}_1 + \mathbf{i}_2)$ and the ones of the second segment as $\frac{1}{2}(\mathbf{i}_1 + \mathbf{i}_2)$, $\mathbf{i}_2$, and $\mathbf{b}_2$. The weights of the middle Bézier points of a segment are chosen such that the curvatures at end points that are also corners of the patch are interpolated. Fig. 11a illustrates this scheme.

9B Avoiding Overshooting Nonconvex Patch Boundary Curves

We describe a construction for a nonconvex boundary curve that will be represented by two conic segments which both lie entirely inside R. Using a tuple notation to indicate the ordered vertices of a polygon, we use the intersection of the polygon $(\mathbf{c}_0, \frac{1}{2}(\mathbf{c}_0 + \mathbf{d}_0), \frac{1}{2}(\mathbf{c}_1 + \mathbf{d}_1), \mathbf{c}_1)$ and the tangent passing through $\mathbf{v}_i$ as middle Bézier point of the first segment. We construct the middle Bézier point of the second segment analogously. We use the intersection of the line passing through $\mathbf{v}_i$ and $\mathbf{v}_j$ and the line passing through the two middle Bézier points as break point of the two segments. Fig. 11b illustrates this case.

9C Avoiding Overshooting Interior Patch Curves

We assume that one needs to interpolate a tangent vector at only one of the two end points. Furthermore, we assume that the tangent vector to be interpolated is $\mathbf{t}_i$, associated with the end point $\mathbf{v}_i$ (a vertex in the initial, triangular contour approximation), and that the other end point is $\mathbf{m}_{ijk}$. We represent the interior patch curve by one conic segment. Its Bézier points are $\mathbf{b}_0 = \mathbf{v}_i$ and $\mathbf{b}_2 = \mathbf{m}_{ijk}$, and the construction of $\mathbf{b}_1$ is the same as the one described under b).

In the implementation, Cases 1-8 must be dealt with. Case 9 can be used as an option. We recommend to always use the methods described under Case 9 to avoid "overshooting" boundary curves and patches. Furthermore, one must take numerical instabilities into account in order to identify any of these cases.

6 Remarks Regarding an Exact Representation

More research is needed for the determination of boundary planes for conics which are generated inside a cube. In order for a rational quadratic triangular patch to be a quadric, the patch must pass through the intersection of the three boundary planes, see [2]. Thus, it might be useful to arrange the boundary planes (which are currently formed as averages) in a fashion such that the quadric condition is met. An alternative would be to elevate the degree of all patches to degree four, since those allow for the representation of quadrics with arbitrary boundary curves.

It is clear from (1) and (2) that a contour of the trilinear interpolant is a cubic surface due to the existence of the cubic term xyz. In the previous sections, we have described an approximation of a contour in terms of triangular rational-quadratic Bézier patches. Without adaptive subdivision, the errors in this approximation can become very large. Consider the contours $xyz = c$, and let c take on small values. The implicit cubic comes arbitrarily close to the origin, while our approximating rational-quadratic patches will approach the point $\mathbf{p} = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)^T$ as the closest point to the origin. This is illustrated in Fig. 12: The Bézier control polyhedron (broken lines), the patch boundary curves (solid curves), and the point $\mathbf{p}$ are shown. Thus, in this case, the error will never be less than $\frac{1}{3}\sqrt{3}$, and it can only be lowered using subdivision.

One can obtain the exact implicit representation of a boundary curve of a triangular patch by inserting the parametric plane equation of a plane P containing the boundary curve into (1) or (2). Representing a point $\mathbf{x}$ in P as $\mathbf{x}(u, v) = \mathbf{x}_0 + u\mathbf{d}_1 + v\mathbf{d}_2$, where $\mathbf{x}_0$ is a point in P and $\mathbf{d}_1$ and $\mathbf{d}_2$ are two basis vectors in P, and inserting $\mathbf{x}(u, v)$ into (1) or (2), one obtains an implicit cubic boundary curve representation. The ten cubic terms appearing in this representation are $u^i v^j$, $i, j \geq 0$, $i + j \leq 3$. Thus, one can compute the exact implicit representations of the three boundary curves of a triangular patch. Furthermore, one can use well-known implicit Bézier representations for these cubic curves by constructing stencils of 10 Bézier ordinates in each boundary curve plane, see [1], [8], and [25].

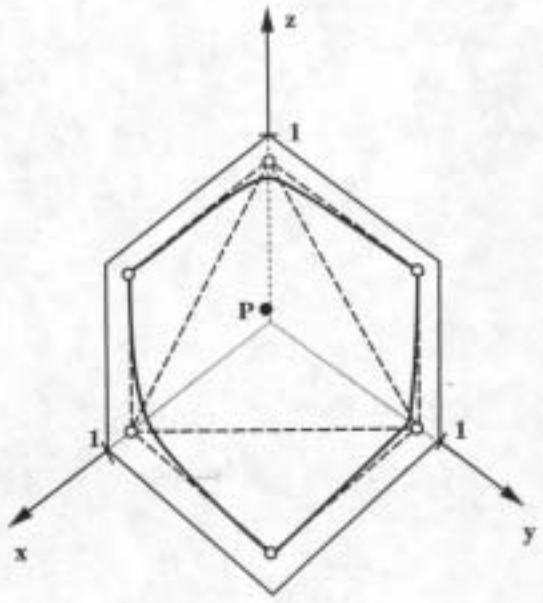

Fig. 12. Contour approximation of $xyz = c$; $x, y, z \in [0, 1]$, $c \ll 0.001$.

The values of the Bézier ordinates in this representation are defined by the gradient and the Hessian (see Appendix) of the known trilinear interpolant at the endpoints of each cubic curve.

Thus, one could construct an implicit representation of all triangular patches. Each triangular patch would be the contour of a cubic polynomial in Bernstein-Bézier form defined over a tetrahedron, using barycentric coordinates. It is beyond the scope of this paper to explore such a construction in detail.

7 Applying the Method to a Rectilinear Grid

Considering a rectilinear grid consisting of multiple cells, one could naively apply the construction described in the previous sections to each cell. Unfortunately, the union of all resulting triangular patches would be a discontinuous surface. The reason for this is the fact that the piecewise trilinear approximation of the given function values F_{ijk} is only C^0-continuous, which means that gradients and Hessians are discontinuous on cell faces. In order to generate a contour approximation that is an overall C^0-continuous surface, we must have unique gradient and Hessian values for each vertex $\mathbf{v}$ in the initial, triangular contour approximation resulting from the marching cubes method. We obtain unique gradient and Hessian values for a particular point $\mathbf{p}$ by averaging the different gradients and Hessians associated with the cells sharing the edge containing $\mathbf{p}$.

8 Examples

The examples shown in Figs. 13-18 were obtained by applying the contour approximation technique to analytically defined functions and computerized axial tomography (CAT) data. The analytical functions used for the generation of Figs. 13-15 were evaluated on a uniform rectilinear grid using the same number of x, y, and z values in all three coordinate directions. The CAT data used for the generation of Figs. 16-18 were provided as stacks of slices, each

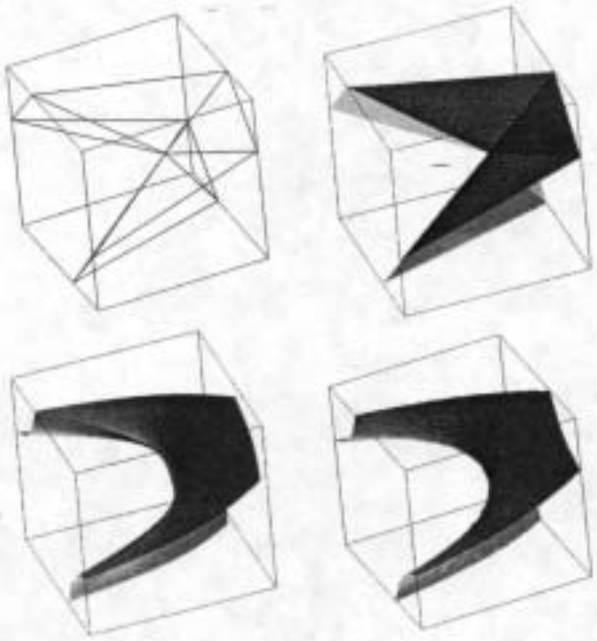

Fig. 13. Triangular approximation, rational-quadratic approximation, and exact contour of trilinear function defined by the function values $f_{0,0,0} = 0$, $f_{1,0,0} = 0$, $f_{0,1,0} = 1$, $f_{1,1,0} = 0.5$, $f_{0,0,1} = 1$, $f_{1,0,1} = 0.5$, $f_{0,1,1} = 0$, and $f_{1,1,1} = 0$; contour value: 0.3333; resolution: $2 \times 2 \times 2$ grid points.

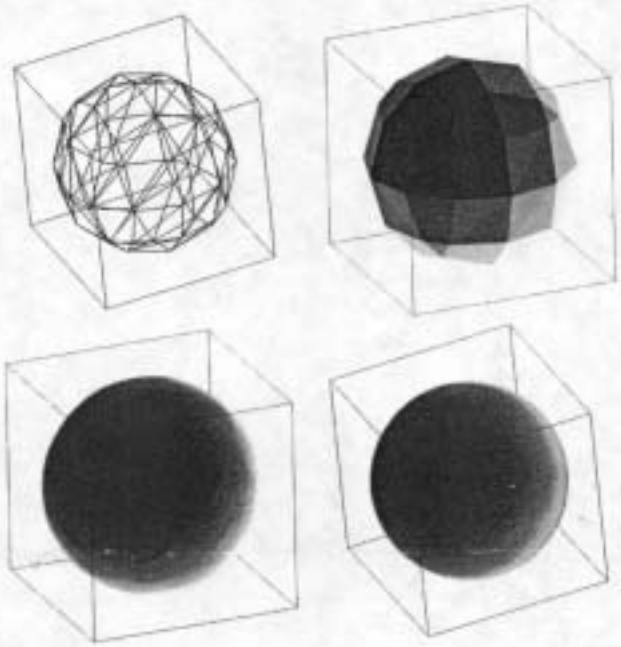

Fig. 14. Triangular approximation, rational-quadratic approximation, and exact contour of trilinear function defined by the function values $f_{0,0,0} = 0.2$, $f_{1,0,0} = 0.9$, $f_{0,1,0} = 0.7$, $f_{1,1,0} = 0.0$, $f_{0,0,1} = 0.9$, $f_{1,0,1} = 0.2$, $f_{0,1,1} = 0.1$, and $f_{1,1,1} = 0.9$; contour value: 0.4804; resolution: $2 \times 2 \times 2$ grid points.

slice being discretized by a uniform rectilinear grid and the spacing in axial direction being uniform.

Figs. 13-15 show the initial triangular approximations (upper-left parts, wire frame renderings, and upper-right parts, flat-shaded renderings), the piecewise rational-quadratic approximations (lower-left parts, flat-shaded renderings), and the exact contours (lower-right parts,

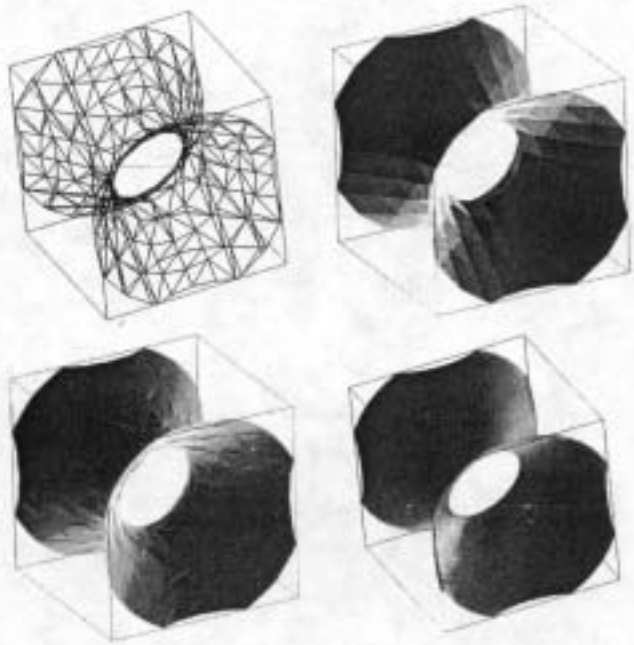

Fig. 15. Triangular approximation, rational-quadratic approximation, and exact contour of $f(x, y, z) = 0.01\ ((x-0.5)^2 - (y-0.5)^2 + (z-0.5)^2$; contour value: 0.2; resolution: $9 \times 9 \times 9$ grid points.

Gouraud-shaded renderings). Figs. 16-18 only show the initial triangular approximations (left parts, flat-shaded renderings) and the piecewise rational-quadratic approximations (right parts, Gouraud-shaded renderings). For Figs. 13-17, the initial triangular approximations were obtained by computing points on the contours along the edges of the grid cells, connecting these points, and triangulating the resulting closed polygons. The initial triangular approximation of the skull shown in Fig. 18 consists of only 5,277 triangles that were obtained by applying the data reduction scheme described in [14] to the initial triangular approximation of the skull shown in Fig. 17.

The piecewise rational-quadratic approximations were obtained by applying the approximation scheme described in this paper to the initial triangular approximations. Each rational-quadratic patch was discretized by triangles, which were flat-shaded. The renderings of the exact contours, shown in Figs. 13-15 only, were obtained by using a high-resolution uniform rectilinear grid for the discretizations of the underlying, known functions $f(x, y, z)$, $x, y, z \in [0, 1]$, applying the MC method described in [21], and using Gouraud-shading for rendering.

9 Conclusions

We have described a method for the approximation of a contour of the trilinear interpolant by triangular rational-quadratic Bézier patches. Each patch is a rational-quadratic surface that locally approximates the contour.

Future research will address the reduction of the number of triangular patches resulting from this method and using the triangular patches for interactive modeling applications, including potential degree raising and subdivision schemes for defining approximations with higher-order continuity. Another interesting aspect to be investigated is

Fig. 16. Triangular approximation and rational-quadratic approximation of hip; triangular approximation consisting of 25,256 triangles.

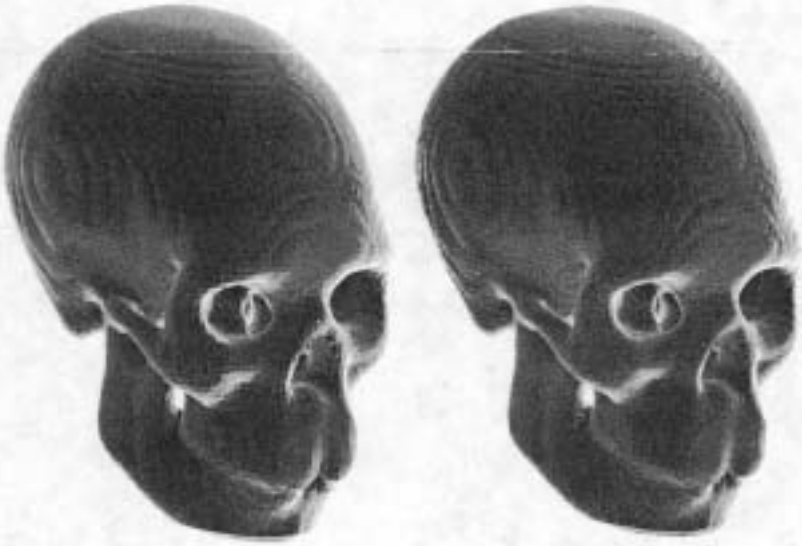

Fig. 17. Triangular approximation and rational-quadratic approximation of skull; triangular approximation consisting of 52,731 triangles.

Fig. 18. Reduced triangular approximation of skull and rational-quadratic approximation; triangular approximation consisting of 5,277 triangles.

the automatic movement of the Bézier control points of the triangular rational-quadratic patches in order to decrease tangent plane discontinuities along patch boundaries.

Appendix

The normal curvature at a point $\mathbf{x} = (x, y, z)^T$ on a contour of the trilinear interpolant is computed based on a lemma in [6], [7].

LEMMA. *Let $f(\mathbf{x}) = f(x, y, z) = T$ be a contour. If $\mathbf{t}$ is a unit tangent vector of the contour at $\mathbf{x}$, then the normal curvature of the contour in the direction of $\mathbf{t}$ is*

$$\kappa(\mathbf{t}) = -\frac{\mathbf{t}^T H \mathbf{t}}{\|\nabla f\|}, \tag{15}$$

where

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right) \tag{16}$$

is the gradient of f at $\mathbf{x}$, $\|\nabla f\|$ is the gradient's length, and

$$H = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2} \end{pmatrix} \tag{17}$$

is the Hessian matrix of f at $\mathbf{x}$.

PROOF. See [6], [7]. □

Computing the Hessian of the trilinear interpolant (1) yields

$$H = \begin{pmatrix} 0 & \Delta^{1,1,0} + \Delta^{1,1,1} z & \Delta^{1,0,1} + \Delta^{1,1,1} y \\ \Delta^{1,1,0} + \Delta^{1,1,1} z & 0 & \Delta^{0,1,1} + \Delta^{1,1,1} x \\ \Delta^{1,0,1} + \Delta^{1,1,1} y & \Delta^{0,1,1} + \Delta^{1,1,1} x & 0 \end{pmatrix}. \tag{18}$$

This lemma and *Meusnier's Theorem*, see [10], are used to compute the exact curvatures at $\mathbf{v}_i$ and $\mathbf{v}_j$ of the patch boundary curve that passes through these points and lies on the associated contour of the trilinear interpolant and in the plane P. In general, the plane P, used for the construction of patch boundary curves in the interior of a cell, does not define normal sections at $\mathbf{v}_i$ and $\mathbf{v}_j$ since P does not necessarily contain the gradients of the trilinear interpolant at these two points. According to *Meusnier's Theorem*, the curvatures of planar sections at these two points are the same as long as the tangent directions considered are the same. Thus, the plane P does not have to define a normal section.

For our purposes, we compute the exact curvatures at $\mathbf{v}_i$ in the direction of $\mathbf{t}_i$ and at $\mathbf{v}_j$ in the direction of $\mathbf{t}_j$. The directions are defined by the plane P. We refer the interested reader to the differential geometry literature for more detail on the geometrical principles we are utilizing in our constructions.

ACKNOWLEDGMENTS

This work was supported by the U.S. National Science Foundation under contracts ASC-9210439 (Research Initiation Award) and ASC 9624034 (CAREER Award) to Mississippi State University and the University of California, Davis. In addition, this project was funded by the U.S. Office of Naval Research under contract N00014-97-1-0222 and the U.S. Army Research Office under contract ARO 36598-MA-RIP to the University of California, Davis.

We thank the members of the Visualization Thrust at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis, for their help.

REFERENCES

[1] C.L. Bajaj, J. Chen, and G. Xu, "Modeling with Cubic A-Patches," *ACM Trans. Graphics*, vol. 14, no. 2, pp. 103-133, 1995.

[2] W. Boehm and D.C. Hansford, "Bézier Patches on Quadrics," *NURBS for Curve and Surface Design*, G. Farin, ed., pp. 1-14. Philadelphia, Penn: SIAM, 1991.

[3] W. Boehm and D.C. Hansford, "Parametric Representation of Quadric Surfaces, *Modélisation Mathématique et Analysis Numérique*, vol. 26, pp. 191-200, 1991.

[4] W. Boehm and H. Prautzsch, *Geometric Concepts for Geometric Design*. Wellesley, Mass: A.K. Peters, Ltd., 1994.

[5] B. K. Choi, H.Y. Shin, Y.I. Yoon, and J.W. Lee, "Triangulation of Scattered Data in 3D Space," *Computer-Aided Design*, vol. 20, pp. 239-248, 1988.

[6] J.H. Chuang and C.M. Hoffmann, "Curvature Computations on Surfaces in *n*-Space," Technical Report #90.2, The Leonardo Fibonacci Inst. for the Foundations of Computer Science, Trento, Italy, 1990.

[7] J.H. Chuang and C.M. Hoffmann, "Curvature Computations on Surfaces in *n*-Space," *Modélisation Mathématique et Analysis Numérique*, vol. 26, pp. 95-112, 1992.

[8] W. Dahmen, "Smooth Piecewise Quadric Surfaces," *Mathematical Methods in Computer Aided Geometric Design*, T. Lyche, and L.L. Schumaker, eds., pp. 181-193. Boston: Academic Press, 1989.

[9] G. Farin, "Triangular Bernstein-Bézier Patches," *Computer Aided Geometric Design*, vol. 3, pp. 83-128, 1986.

[10] G. Farin, *NURB Curves and Surfaces*. Wellesley, Mass.: A.K. Peters, Ltd., 1995.

[11] G. Farin, *Curves and Surfaces for CAGD*, 4th ed. Boston: Academic Press, 1997.

[12] C.M. Grimm and J.F. Hughes, *Smooth Isosurface Approximation*, B. Wyvill, and M.P. Gascuel, eds. *Implicit Surfaces '95*, Eurographics workshop, pp. 57-76. U.K.: Blackwell Publishers, 1995.

[13] B. Hamann, "Modeling Contours of Trivariate Data, *Modélisation Mathématique et Analysis Numérique*, vol. 26, pp. 51-75, 1992.

[14] B. Hamann, "A Data Reduction Scheme for Triangulated Surfaces," *Computer Aided Geometric Design*, vol. 11, no. 2, pp. 197-214 1994.

[15] B.Hamann, G. Farin, and G.M. Nielson, "A Parametric Triangular Patch Based on Generalized Conics," *NURBS for Curve and Surface Design*, G. Farin, ed., pp. 75-85. Philadelphia, Penn.: SIAM, 1991.

[16] D.C. Hansford, "Boundary Curves with Quadric Precision for a Tangent Continuous Scattered Data Interpolant," PhD dissertation, Arizona State Univ., Tempe, AZ, 1991.

[17] D.C. Hansford, R.E. Barnhill, and G. Farin, "Curves with Quadric Boundary Precision," *Computer Aided Geometric Design*, vol. 11, pp. 519-531, 1994.

[18] C.L. Lawson, "Software for C^1 Surface Interpolation," *Mathematical Software III*, J.R. Rice, ed., pp. 161-194. San Diego, Calif: Academic Press, 1977.

[19] W.E. Lorensen and H.E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *Computer Graphics*, vol. 21, pp. 163-169, 1987.

[20] J. Niebuhr, "B-Patches on Quadrics (in German)," PhD dissertation, Technische Universität Braunschweig, Germany, 1991.

[21] G.M. Nielson and B. Hamann, "The Asymptotic Decider: Resolving the Ambiguity in Marching Cubes," *Proc. Visualization '91*, G.M. Nielson, and L.J. Rosenblum, eds., pp. 83-91, 1991.

[22] L. Piegl and W. Tiller, *The NURBS Book*. New York: Springer-Verlag, 1995.

[23] T. Schreiber, "Arithmetische Operationen auf Bézierflächen," Internal Report 224/92, Fachbereich Informatik, Technische Universität Kaiserslautern, Germany, 1992.

[24] L.L. Schumaker, "Computing Optimal Triangulations Using Simulated Annealing," *Computer Aided Geometric Design*, vol. 10, pp. 329-345, 1993.

[25] T.W. Sederberg, "Piecewise Algebraic Surface Patches," *Computer Aided Geometric Design*, vol. 2, nos. 1-3, pp. 53-59, 1985.

Bernd Hamann received a BS in computer science, a BS in mathematics, and an MS in computer science from the Technical University of Braunschweig, Germany. He received his PhD in computer science from Arizona State University in 1991. From 1991 to 1995, he was a faculty member in the Department of Computer Science at Mississippi State University and a research faculty member at the U.S. National Science Foundation Engineering Research Center for Computational Field Simulation. Dr. Hamann was awarded a 1992 Research Initiation Award by Mississippi State University, a 1992 Research Initiation Award by the U.S. National Science Foundation, and a 1996 CAREER Award by the U.S. National Science Foundation. In 1995, he received a Hearin-Hess Distinguished Professorship in Engineering by the College of Engineering, Mississippi State University. Dr. Hamann is currently an associate professor of computer science and codirector of the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis, and an adjunct professor of computer science at Mississippi State University.

His primary interests are visualization, computer graphics, and computer-aided geometric design (CAGD). He is the author of numerous publications and has presented his research at leading conferences in the U.S. and in Europe. He is a member of the ACM, IEEE, and SIAM.

Issac J. Trotts is an undergraduate student in computer science at the University of California, Davis. His research interests are computer graphics and scientific visualization. He is a member of the ACM.

Gerald E. Farin received a BS, an MS, and, in 1979, a PhD in mathematics from the Technical University of Braunschweig, Germany. Dr. Farin is a professor of computer science and engineering at Arizona State University. His primary interests are computer-aided geometric design (CAGD) and, in particular, nonuniform rational B-splines (NURBS). He is the author of numerous publications and the author of one of the most prominent textbooks in CAGD, *Curves and Surfaces for CAGD—A Practical Guide*. Dr. Farin is a member of SIAM, for which he is currently serving as secretary of the Activity Group on Geometric Design.